Simulation of a network capacity market and three middle-man strategies t o price and sell dynamically routed point-to-point connections

Lars Rasmusson, Erik Aurell
Swedish Institute of Computer Science
Box 1263, S-164 29 KISTA, SWEDEN

20th November 2002

.In a simulation of a computer network, the capacity between pairs of border routers in a network domain is sold on a spot market. End-users establish point-to-point connections across several domains by buying capacity in the domains along a path in the network. This paper compares three different trading strategies: spot requests, null broker requests, and derivative broker requests. Their performance is measured in terms of the ratio of successful connections, and the cost to establish a connection. Simulations of a network results show that the derivative broker requests typically performs better than the other two, especially in networks where prices fluctuate rapidly.

Report: T2002:23
ISRN: SICS-T–2002/23-SE
ISSN : 1100-3154

Simulation of a network capacity market and three middle-man strategies to price and sell dynamically routed point-to-point connections

Lars Rasmusson, Erik Aurell

20th November 2002

Abstract

In a simulation of a computer network, the capacity between pairs of border routers in a network domain is sold on a spot market. End-users establish point-to-point connections across several domains by buying capacity in the domains along a path in the network. This paper compares three different trading strategies: spot requests, null broker requests, and derivative broker requests. Their performance is measured in terms of the ratio of successful connections, and the cost to establish a connection. Simulations of a network results show that the derivative broker requests typically performs better than the other two, especially in networks where prices fluctuate rapidly.

Keywords: computer networks, bandwidth markets, QoS, combinatorial trading, option pricing

1 Introduction

Computer networks are now being used to transfer live video and other data that need reliable service from the network, in terms of latency, jitter, and loss [1, 2, 3]. Bandwidth or *capacity markets* can help to provide the necessary end-user quality of service guarantees [4, 5, 6, 7]. Capacity trading enables users to reserve capacity in congested networks, and thereby get a guaranteed throughput, even when the network is congested.

The networking QoS routing deals with routing under path constrains. Since it is NP-hard to find one optimal *multi-constrained path* that satisfies a set of (more than one) constraints, approximative solutions such as hierarchical routing [3, 8], or relaxing multi-constraint to single constraints [9] are necessary.

Most approaches to combinatorial allocation use a centralized distributor that solves a global constraint problem c.f. [5, 10, 4, 6, 11, 12, 13, 7], which has the obvious disadvantage in a network scenario that, besides solving the allocation problem, the distributor must first collect all demands, and then distribute all allocation information before the system's state is updated.

Another relaxation of the QoS problem is to only give statistical service guarantees [14, 15, 16, 17, 18]. This is in the spirit of service models such as DiffServ [19] that only aims to provide QoS within a centrally managed network domain, while it fails to provide guarantees along an entire path.

In this paper, techniques and concepts borrowed from the financial markets play an important part. The most closely related previous literature on derivative pricing in relation to computer networks deal with trading in just one single asset, *network access,* [20, 21], without addressing the combinatorial pricing problem. Combinatorial problems are addressed in [22, 23, 24], where forward contracts on time-slotted network capacity are priced. This has the disadvantage of producing a large number of fundamental assets, with interrelations due to both network topology and time.

The structure of the rest of this paper is as follows: section 2 contains a short background to network services implemented in terms of traded capacity, and section 3 presents the model of the network and user demand. In section 4 we give the results of the simulations in the form of hypothesis testing. Section 5 concludes the paper with a discussion of the results.

2 Background

In the bandwidth market model used for the simulation results reported in this paper, the trading relies on a network admission mechanism that controls the end-users' access to the network recently discussed in [25]. End-users buy *capacity tokens* on the capacity markets. The tokens are used by the admission mechanism to verify that a user has all the necessary capacity, from source to destination, before his traffic is let out on the network. To provide shaping of each flow, the admission mechanism uses trusted access

nodes that shape the traffic only at the network edges. The edge admission mechanism seems to be necessary in a real network, since it would be too costly to shape individual flows in the network core. We do however leave the door open for the possibility of future developments in distributed shaping mechanisms. The simulation described in this paper only models the capacity trading, and it is assumed that some admission mechanism is in place.

The motivation for this particular kind of market-based network QoS provision and network architecture is put forward in [26], where also the market dynamics is described. The idea to price *network services* in terms of financial derivative contracts of the spot market assets was introduced in [27].

2.1 Network services

A network service can be interpreted as the right to own capacity in certain subnetworks at certain time points. For instance, *a backup* service can consist of a fixed capacity on a path between the server location and the backup storage location for one hour every night. A *video service* can consist of access to one of several movie repositories for three hours daily. In this paper, we consider the *point-to-point* service, which consists of the capacity on one of the several paths between two places, with specified start and end times.

The fundamental problem in producing a point-to-point service is as follows. A network is represented by graph with nodes and edges, where the nodes correspond to the congested resources. To send traffic between two nodes, a user must have acquired capacity in *every* node along *some* path between the two nodes. It is not sufficient to only have capacity in some nodes on a path. Nor is it useful to have capacity in nodes that are not on the chosen path. The user has a limited budget which sets the maximal cost that he is willing to pay for the connection. Furthermore, an end-user may know its capacity demand some time in advance, which gives him the opportunity to buy different pieces of the capacity at different times before the *expiration time* of the demand. Thus the end-user faces a complicated problem of deciding which resources to buy, and when.

2.2 Trading strategies

In this paper we simulate three different trading strategies, and compare them with each other in the following way. In the simulation, end-users generate

a flow of requests for point-to-point connections in the network. There are two different types of requests,

- spot requests; the user waits until the expiration time, and then buys the path that is cheapest for the moment, and
- broker requests; the user immediately asks a broker to supply the capacity on some path at the expiration time.

By using brokers, end-users do not have to do any trading of their own. All the risk involved in trading is moved from the user to the broker. The spot requests are the null hypothesis to the broker requests. Brokers are useful if they result in more successful requests than spot requests do, and/or if the admission cost is lower for broker requests.

There are two types of broker strategies, depending on the type of broker request,

- null broker requests; the broker only aggregates a number of request, and only trade when the requests expire, and
- derivative broker requests; the broker computes the value of the contingent claim on the resources requested by the end-user, and sells a *derivative contract* to the user. Between the request time and the expiration time, the broker buys and sells capacity to reduce the risk of loss due to price fluctuations, and at the expiration time, the broker buys the remaining required resources.

The null brokers are the null hypothesis to the derivative brokers. If all the profit from the derivative brokers is due to the fact that they aggregate a number of risky demands, then the null broker and the derivative broker will perform equally well. If the derivative broker performs better than the null broker, the increased performance is due to the hedging strategy.

The requests have a maximal budget constraint that determines whether the request will be accepted or rejected. The admission performance of the request types is measured as the fraction of rejected requests. The net cost per request for the user and for the broker determines how well the pricing mechanism captures the capacity price movement. These metrics are studied in section 4.

The main topic of interest in this paper is the derivative broker request. Derivative brokers accept early requests, and hedges the request until the starting time by rebalancing a portfolio so that the expected value of the total portfolio is kept at zero. Spot requests and null broker requests are null hypothesis request types. Spot requests model network users in a network that does not have derivatives to price services. With spot requests, the users must do all the trading, take all the risk, and cannot trade in future demand. Null broker requests model a network where requests simply are aggregated, and no hedging is performed by the broker. The risk is transfered to the null broker, who remains passive until the starting time, at which the null broker decides to buy whatever looks cheapest at the moment.

3 Model

3.1 The network

The network is a fixed connected finite undirected graph G. The nodes $V(G)$ denote the network resources, i.e. a congested subnetwork, and the edges $E(G)$ denote the links. Individual nodes are denoted by an index $V_i = V_i(G) \in V(G)$, $i = 1, ..., n$. G' is a *subgraph* of G if $E(G') \subseteq E(G)$ and $V(G') \subseteq V(G)$.

A *network path* is a connected subgraph of G such that

$$\max_{u \in V(G')} \deg_{E(G')}(u) \leq 2.$$

The nodes with degree 1 are called *end-points*. The *network path node set*

$$\text{pathnodes}(s, r) = \{V(x) : x \text{ is a network path with endpoints } s, r\}$$

is the set of node sets on paths between s and r. The *alternative network path node sets* or *alternative paths* $\mathcal{A}(s, r)$ is

$$\mathcal{A}(s, r) \subseteq \text{pathnodes(s,r)}, \text{ such that } \forall a, b \in \mathcal{A}(s, r) : a \neq b \Leftrightarrow a \nsubseteq b$$

$\mathcal{A}(s, r)$ is the largest set of sets of network path nodes with end-points s and r, such that no path has a node set that is a superset of the nodes in another path in the same alternative paths set. The paths in a set are enumerable, and path i is denoted $\mathcal{A}_i \in \mathcal{A}(s, r)$.

For a given pair of end-points s and r, the *resource matrix*

$$v_{ij} = 1 \text{ if } V_j \in \mathcal{A}_i(s,r), \text{ and zero otherwise.} \tag{1}$$

The average number of vertices in the alternative paths is denoted

$$\eta = \frac{1}{|\mathcal{A}(s,r)|} \sum_{i=1}^{|\mathcal{A}(s,r)|} |\mathcal{A}_i|$$

and thus the *a priori* probability of a node belonging on a path between a uniformly chosen pair of nodes is $\frac{\eta}{|V(G)|}$.

3.2 Capacity prices

A network user that owns capacity in a node is entitled to send through the node at a rate less than or equal to the owned amount. Shares of the total capacity in a node is traded on a market.

In the abstract, one can consider any kind of market mechanism, and any type of resulting price processes. In practice, it is important that the market is set up to be as fast as possible, and to have minimal communication and negotiation overhead. In the mechanism used here, the market is run by a market maker that adjusts the prices in response to the changing capacity demand. The market maker only accepts *market bids*, i.e. bids to trade a certain quantity at the market price, whatever it may be. The bid is a buy bid if the quantity is positive, and a sell bid if the quantity is negative. The market does not accept *limit bids*, i.e. bids that only are valid if the price is within a certain range, or other conditional bids. Since there is no order book of unfilled limit bids, the market overhead is kept minimal.

The market is cleared at regular times, $t = 1, 2, ...$ The spot price for one unit of capacity in node $j \in V$ is adjusted according to

$$S_j(t) = \mathrm{S}_j(0) \exp \frac{\omega_j(t)}{\lambda_j} \tag{2}$$

where $\omega_j(t)$ is the total demand for capacity in node j at time t. All unfilled bids get to trade at price $S_j(t)$. The model parameter λ_j, also called the *liquidity parameter,* sets the amount of capacity in the node. A small amount of capacity is modeled with a small λ, which results in large price fluctuations, and vice versa.

Besides paying for buying capacity, a capacity owner pays a *holding fee* $\epsilon S_j(t)$ for holding the capacity in the interval $[t, t+1)$.

3.3 Requests

Network users generate requests to the system. The requests arrive as a Poisson process, with intensity $\frac{1}{\mu}$, where μ is the average number of arrivals per unit of time.

Request i asks to buy capacity in all the vertices in any of the network paths in the alternative path set $\mathcal{A}(s_i, r_i)$. The source and destination nodes s_i and r_i are drawn uniformly from $V(G)$. Request i asks to hold the capacity in the nodes $d_i \in Ge_+[\frac{1}{d}]$ time intervals, where d is the average duration, and $Ge_+(p)$ denotes the *first time* distribution.

Request i has a maximal budget $c_i \in Exp[\frac{1}{c}]$, where c is the average budget. When the request arrives, an *admittance cost* is computed. The admittance cost is the user's or broker's estimate of the cost to produce the service. If the admittance cost is less than the budget, the request is *accepted*, otherwise it is *rejected.*

The *request type* determines how the admittance cost is computed. There are three types of requests, *spot requests, null broker requests,* and *derivative broker requests.* In a simulation, $q \in [0, 1]$ of the requests are broker requests, and $(1 - q)$ of the requests are spot requests. Derivative brokers and null brokers are not mixed, thus a simulation contains either only spot and derivative broker requests, or only spot and null broker requests.

Spot requests are traded immediately, while broker requests arrive $a_i \in Ge_+(\frac{1}{a})$ time intervals before the capacity is actually needed (the expiration time), where a is the mean *alert time.*

For spot requests, the admittance cost is

$$c_{spot}=\{v\Big(S(t) - \bar{S}(t) + d_i\epsilon\bar{S}(t)\Big)\}_k \tag{3}$$

where

$$k = \text{argmin}_j\{vS(t)\}_j \tag{4}$$

is the index of the cheapest path, based on the price at t, $\bar{S}(t)$ is the average cost taken over an interval $(t - \tau, t]$, and v is the demand's resource matrix (see eq. 1). The spot cost is the user's estimate of the total cost of realizing the service. In the end, the user may have to pay more or less than c_{spot}.

For *null broker requests*, the admittance cost is

$$c_{null} = d_i\epsilon \min_j\{vS(t)\}_j \tag{5}$$

For *derivative broker requests,* the admittance cost is computed as a call option to buy the cheapest path at the *starting time* $t + a_i$ and give back the resources at the *ending time* $t + a_i + d_i$. The send fee is payed by the broker, and is therefore covered by the option price that the network user pays once.

$$c_{deriv} = f(t, v, t + a_i, d_i, K) + K \tag{6}$$

where the strike price is put to

$$K = \min(\frac{c_i}{2}, \min_j \{vS(t)\}_j).$$

Users that use broker requests are guaranteed to only pay the admittance cost, while the broker may have to pay more or less, due to price fluctuation.

In the simulation, Monte-Carlo simulation was used to compute the derivative price

$$f(t, S(t), v, T, d, K) = E^Q[\max(\min_k \sum_j v_{kj} C_j - K, 0) | S(t)]. \tag{7}$$

Here $T > t$ is the starting time of the request, Q is the risk-neutral measure, and

$$C_j = S(T) + \epsilon \int_T^{T+d_i} S(t)dt - S(T + d_i)$$

is the cost of buying capacity j at T, holding it from T to $T + d_i$, and selling it at $T + d_i$. The option price is evaluated with 100000 runs of Monte Carlo in a procedure as described in [28]. In the simulation, we put the risk free rate $r = 0$, which gives S the dynamics $dS_j = S_j\{\sigma dW\}_j$ under Q. dS and dW are $|V|$-dimensional processes, $W(t)$ being a Wiener process with i.i.d. components. The price diffusion covariance σ^2 is computed using the estimator for a mean-reverting process with multiplicative diffusion derived in [26]. The hedge

$$h_{ij} = -\frac{\partial f}{\partial S_j}$$

is computed at the same time, using the same simulation trajectory. The elements of the hedge h were constrained to be within $[-2, 2]$, to prevent instability in the hedge composition due to numerical errors from the Monte Carlo simulation.

3.4 Parameters

Simply varying the parameters μ, d, a, and c, does not allow us to compare the system performance directly, as one will compare systems with different units for currency, time, and capacity with each other. Instead, we will vary the intrinsic dimensionless parameters that govern the system behavior, and scale the other parameters accordingly.

The arrival rate μ sets the intrinsic time scale of the simulation, as it determines the number of requests that arrive per time unit.

The system is a $M/M/\infty$ queuing system with arrival rate μ and departure rate $\frac{1}{d}$. If all requests are accepted, the offered load

$$L_t \in Po[\mu d(1 - e^{-t/d})]$$

if $L_0 = 0$. Let the parameter

$$L = \lim_{t \to \infty} \mathbb{E}[L_t] = \mu d$$

denote the *characteristic load* which is the asymptotic offered load, i.e. the average number accepted and active requests at one time.

The expected number of offered requests *per link* is

$$\bar{\omega} = L \frac{\eta}{|V(G)|} \propto L,$$

since $\frac{\eta}{|V(G)|}$ is a constant for a given network. The prices are scaled so that the price at load $\bar{\omega}$ is S_L^*. S_L^* is called the *scaled characteristic price*. The scaling is achieved by setting the price $S(0)$, i.e. the price at time zero, when the load is zero. Since

$$S_L^* = S(0)e^{\frac{L}{\lambda}}$$

the baseline price $S(0)$ is set to

$$S_j(0) = S_L^* e^{-\frac{L}{\lambda_j}}.$$

Thus, the simulation runs with scaled prices that vary approximately around S_L^*, while the *natural prices* are proportional to $e^{L/\lambda} S(t)$.

The scale of the price processes determines the scale of the budget size. The average budget

$$c = \rho S_L^*$$

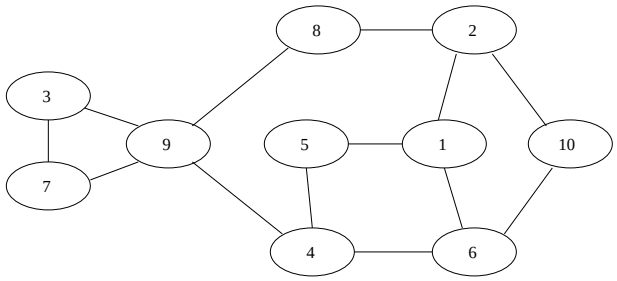

Figure 1: The topology of the randomly generated network used in the simulation. For example, there is only one *alternative path* from node 2 to 10, but three *alternative paths* from node 2 to 6.

is controlled by the dimensionless budget parameter ρ.

In the simulation, all network links have the same capacity, thus we set $\lambda_j = \lambda$ for all $j \in V$.

The network, with the topology shown in fig. 1 has $n = 10$ nodes. It is the same network topology used in the bandwidth market in [26]. The 212 alternative paths in the network have average path length $\eta = \frac{854}{212} \approx 4$.

3.5 Simulation

One simulation is run for 1000 time steps, $t \in [0, 999]$. In each time step, the net change in demand is computed, and the prices are updated. There are five sources for change in demand:

1. new spot requests that arrive
2. new broker requests that arrive
3. changes in the broker's portfolio due to hedging
4. broker requests that are at their starting time
5. requests that are on their ending time

As we are interested in the performance of the demand types, only the the aggregated behavior of many brokers and users is modeled. The simulation keeps track of the aggregated demand caused by broker requests, the *broker*

demand, $\omega^{(B)}(t)$, and the aggregated demand caused by spot requests, the *spot demand,* $\omega^{(S)}(t)$.

To compute the changes in demand, the simulation proceeds as follows at time t

1. New spot requests arrive with intensity $\frac{1}{\mu}(1-q)$. For each new spot request i that arrives at time t
   (a) Compute c_{spot} and the path number k by eq. (3) and (4).
   (b) If $c_{spot} < c_i$, accept the request, increase the spot demand of resource j with v_{kj}, where v_{kj} is the resource matrix in eq. (1)
   (c) If accepted, add the request to the list of running requests.
2. New broker requests arrive with intensity $\frac{1}{\mu}q$. For each new broker request i that arrives at time t
   (a) If the brokers in the simulation are derivative brokers, compute $c_{broker} = c_{deriv}$ by eq. (6), otherwise, compute $c_{broker} = c_{null}$ by eq. (5).
   (b) If $c_{broker} < c_i$, accept the request, and append it in the list of waiting requests.
3. If the brokers in the simulation are derivative brokers, for each request i in the list of waiting requests with start time$>t$
   (a) Compute the new hedge h for request i, where h_{ij} is the amount of resource j that is used in the hedge.
   (b) Increase the broker demand for resource j with $h_{ij} - h'_{ij}$, where h'_{ij} is the previous hedge for request i. Decrease the demand if the quantity is negative.
4. For each request i in the list of waiting requests with start time$=t$
   (a) Compute the path number k by eq. (4). This is the path assigned to the request.
   (b) Increase the broker demand for resource j with $v_{ij} - h_{ij}$. (If the brokers in the simulation are null brokers, h_{ij} will be zero).

(c) Remove the request from the list of waiting requests, and add it to the list of running requests.

5. For each request i in the list of running requests with end time=t

   (a) If i is a spot request, decrease the spot demand for resource j by v_{kj} where k is the path number chosen for request i at an earlier time in step 1.

   (b) If i is a broker request, decrease the broker demand for resource j by v_{kj} where k is the path number chosen for request i at an earlier time in step 4.

When the change in demand has been computed, the book-keeping starts. Spot request costs are logged per request, while broker request costs are aggregated.

1. The new net demand
$$\omega_j(t) = \omega_j^{(B)}(t) + \omega_j^{(S)}(t)$$
is computed, and the new prices $S(t)$ are updated by eq. (2).

2. The users' cost associated with each *new* accepted spot request i is set to
$$\sum_j v_{kj} S_j(t)$$

3. The cost associated with each running spot request i is increased by
$$\epsilon \sum_j v_{kj} S_j(t)$$

4. The cost associated with each ending spot request i is decreased by
$$\sum_j v_{kj} S_j(t)$$

5. The brokers' cost for trading resources is increased by
$$\sum_j (\omega_j^{(B)}(t) - \omega_j^{(B)}(t-1)) S_j(t)$$

6. The brokers' cost for holding resources is increased by

$$\epsilon \sum_j \omega_j^{(B)}(t) S_j(t)$$

7. Finally, new estimates of the price process parameters σ and α are computed, from a price history 100 time steps long.

8. The time is increased by one, and the simulation loop is reiterated.

4 Results

4.1 Hypotheses

Experiments were made to test the following hypotheses:

1. H1: Using derivative broker requests gives a lower fraction of rejected requests, compared to spot requests and null broker requests.

2. H2: The performance of the derivative brokers is due to their hedging strategy. It is not simply a result of aggregating the gains and losses over many requests.

3. H3: Increased alert time decreases the fraction of rejected requests for derivative broker requests, but not for null broker requests.

4. H4: Using derivative broker requests has a dampening effect on the price fluctuations. The effect is not simply a result of aggregating many requests.

The hypotheses were tested for significance by performing a rank test of the expected value in the data sets. Confidence tests were made using the $paired t-test$, which assumes that the data is iid and Gaussian. The assumption is reasonable here, since a large part of the errors are due to Monte Carlo method imprecisions, thus caused by many small independent error sources, and which justifies use of the Central Limit Theorem.

The $t-test$ tests if hypotheses of the type $\mathbb{E}[X] > 0$ are supported by data points $x_1, ..., x_m$ by checking if the null hypothesis $\mathbb{E}[X] = 0$ is less

likely than the confidence, here taken to be C=0.05. It assumes that the data points are drawn from a R.V. X, and tests if

$$Prob[X > x] < C.$$

The quantity to the left is called the $p - value$ of the data. If the confidence criteria cannot be met, then it may or may not be true that $\mathbb{E}[X] = 0$, and the data is said to be inconclusive.

A similar test, the *mean difference test*, is made to investigate if two sets of samples are drawn from random variables with the same expected value. The mean difference test is used to compare the relative performance of the different trading strategies. The tests were made with the Mathematica program [29].

Other tests such as Wilcoxon's Signed Rank test [30] or the Kruskal-Wallis Rank test [31] may also be used. Nonparametric tests such as these may be preferable due to their robustness, i.e. their insensitivity to small departures of the idealized assumptions, on e.g. underlying distributions, but are on the other hand less sensitive.

4.2 Data sets

Four test sets with simulation results were generated. The parameters were fixed in each individual simulation, but varied over the test set:

- In the first set, the parameter L was varied over $L \in \{10, 20, ..., 100\}$. The L parameter controls the number of concurrently running requests that the network will support in its stationary state.

- In the second set, the parameter λ was varied over $\exp \frac{1}{\lambda} \in \{1.02, 1.05, 1.1, 1.2, 1.3\}$. Instead of varying λ directly, we vary x= $\exp \frac{1}{\lambda}$, since $\exp \frac{1}{\lambda}$ has a more intuitive meaning. It is the relative price movement caused by trading one unit of capacity in a node, e.g. $x = 1.1$ means that the price is moved 10% when the net demand is changed one unit. In networks with small capacity, the price fluctuation is large since the demanded volume is large in comparison to the available amount, and thus λ is small and x is large. The inverse holds for large networks.

- In the third set, the parameter a was varied over $a \in \{1, 2, ..., 10\}$. With larger values of a, the brokers have more time to trade to obtain the necessary resources.

- In the fourth set, the parameter ρ was varied over $\rho \in \{1, 2, .., 10\}$. The budget parameter controls the budget size in relation to the characteristic cost. If ρ is small, the average cost of a request is large compared to the average budget, and *vice versa*.

Besides the above parameters, the parameters q and *broker request type* was varied over $q \in \{0.1, 0.5, 0.9\}$, and $type \in \{null, derivative\}$ in all test sets, in order to compare the different request types along the test set parameter axis. The baseline values of the model parameters were set to $L = 50$, $a = 5$, $\rho = 4 \approx \eta$, $\lambda = 1/\log 1.1$, $\mu = 1$, $d = L/\mu$.

Each simulation ran 1000 time steps. In order to let prices settle before derivative request costs were computed, for $t < 200$, q was set to 0, making all requests spot requests. For $t \geq 200$, q was reset to its simulation value, making a fraction q of the requests broker requests of the type *type*, as specified for the simulation.

The fraction of rejected requests, and the costs to accept the requests, were averaged over the time interval $t \in [500, 1000)$. The price variance σ was measured at $t = 1000$.

Figures 2 to 6 compare the derivative broker requests' performance with

- the null broker requests' performance in a similar situation, and
- the spot requests' performance in the same simulation.

In these figures, the averages of the data points are connected with cubic splines to facilitate reading. The color of the line denotes the request type: red is for derivative broker requests; green is for spot requests, and blue is for null broker requests. The line dashing denotes the q value, i.e. the fraction of broker requests to spot requests. Sparse dashed lines show $q = 0.1$, medium dense dashed lines show $q = 0.5$, and dotted lines show $q = 0.9$. Solid lines show unmixed populations. Thus, as an example, $q = 0.1$ means that there are ten percent broker requests and 90 percent spot requests. Curves with similar dashing can therefore be compared with each other.

Since derivative brokers and null brokers are not mixed, a simulation contains either only spot and derivative broker requests, or only spot and null broker requests. Thus, when derivative brokers are compared to null brokers, they have run in separate simulations, independent of each other. When the derivative brokers are compared to spot requests, they have run in the same simulation, competing against each other.

4.3 Rejected requests

In fig. 2 the fraction of rejected requests y is plotted against the parameters L, λ, a, and ρ. In agreement with $H1$ and $H2$, the derivative broker requests result a lower rejection rate than null broker requests, and spot requests. A mean difference test of the derivative broker rejection rate and the null broker rejection rate shows that the mean difference is significant, except for the cases $q = 0.1$, with varying λ or ρ (see the sparse dashed lines, left column, rows two and four). A similar mean difference test with spot rejection rate shows that the mean difference is significant for $q = 0.9$, or for parameter values to the left on the x-axis. In the other regions, the data is inconclusive.

In agreement with $H3$, increasing the alert time a decreases the rejection rate for the derivative broker, but not for the null broker. It appears that changing a has an effect on the spot requests. This effect is a result of the implicit interaction with the derivative brokers that run in the same simulation. When spot requests from a simulation with null brokers are plotted, no effect due to a is found (not shown in the figures). Thus, the spot requests profit from the derivative brokers ability to trade future demand.

The result of modifying λ is at first counter-intuitive, since the rejection rate decreases with increasing $x = \exp \frac{1}{\lambda}$. This can be explained by noting that large λ (and small x) keep prices very near S^*, where the rejection rate will be on the order of 0.5 (for $\rho = \eta$), while for small λ, prices will mostly deviate significantly from S^*, most of the time remaining below S^*, with occasional large price peaks.

The graphs show that derivative broker reject rate decreases with increasing q, since red curves with denser dashing are below more sparsely dashed curves. In other works, the more derivative brokers the better for the request acceptance rate.

4.4 End-user costs

In fig. 3 the end-users' cost per request is plotted against the parameters L, λ, a, and on the characteristic price scale in fig. 3, and on the natural scale (i.e. rescaled by $\exp \frac{L}{\lambda}$) in fig. 4. The plot of prices on the natural scale shows that the prices increase exponentially with the load.

The prices displayed are the average costs for the accepted requests. For low values of L and λ, the derivative broker requests' average costs are higher than for spot and null broker requests. This is an effect of the high rejection

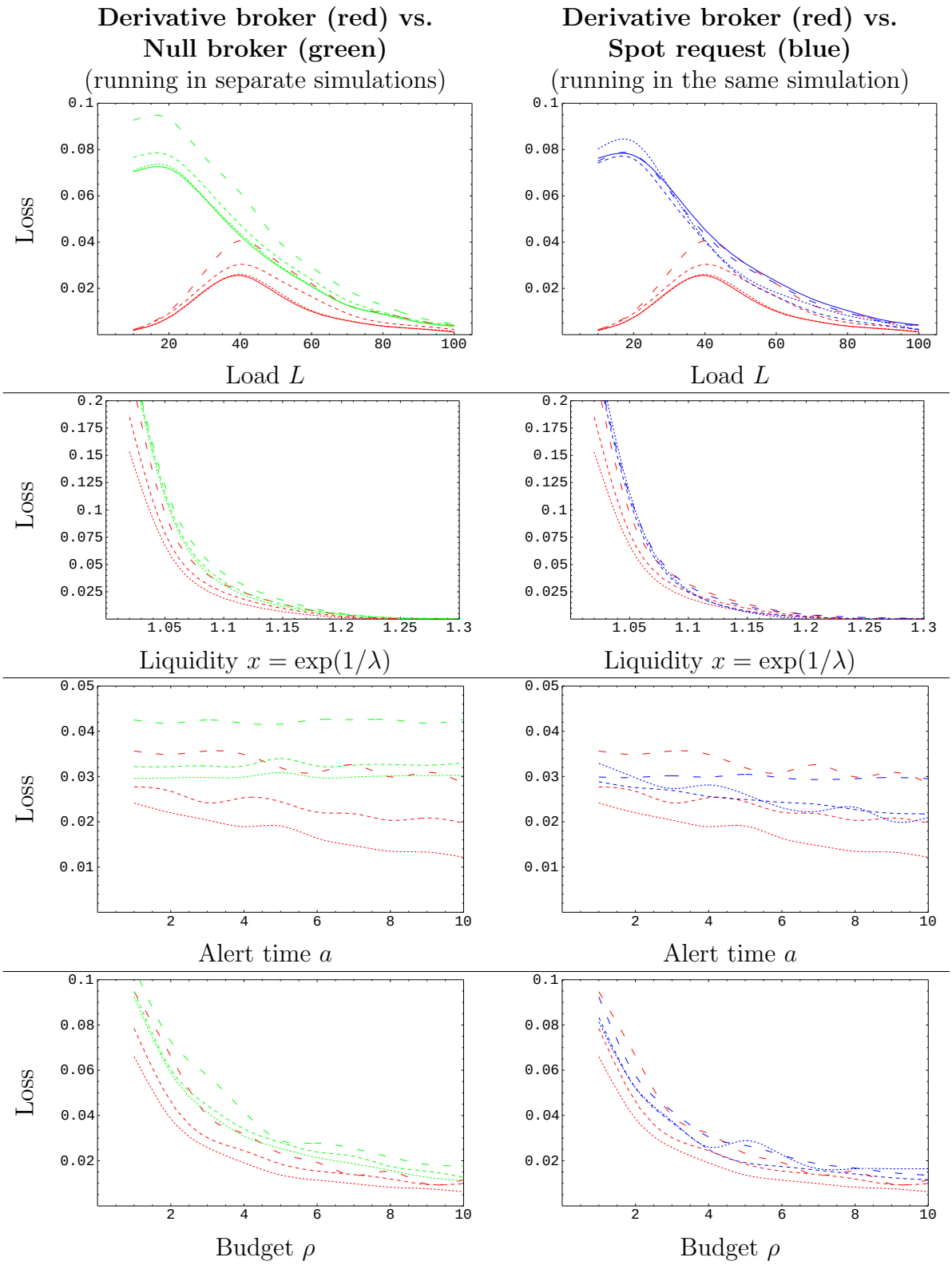

Figure 2: The fraction of rejected requests, or loss, plotted for the four data sets. Color denotes request type, and dashing fraction of broker requests, as explained in sec. 4.2. The horizontal axis on row one shows the load parameter L, on row two, the market liquidity $x = \exp \frac{1}{\lambda}$, on row three, the alert time parameter a, and on row four, the budget size parameter ρ.

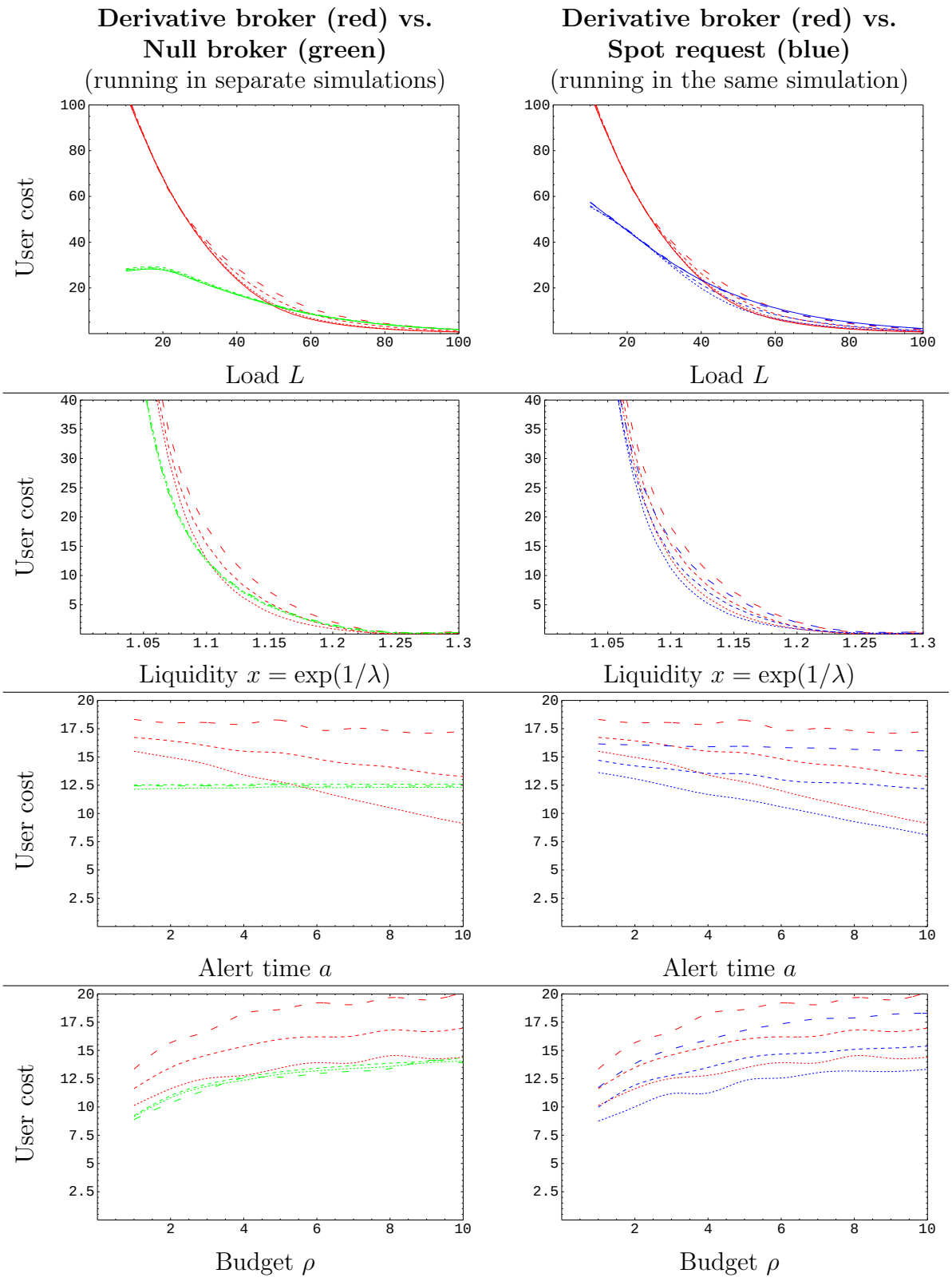

Figure 3: The users' cost per request is plotted on the characteristic cost S^* scale, for the four data sets, varying L, λ, a, and ρ on respective rows. Color and dashing has same meaning as in fig. 2.

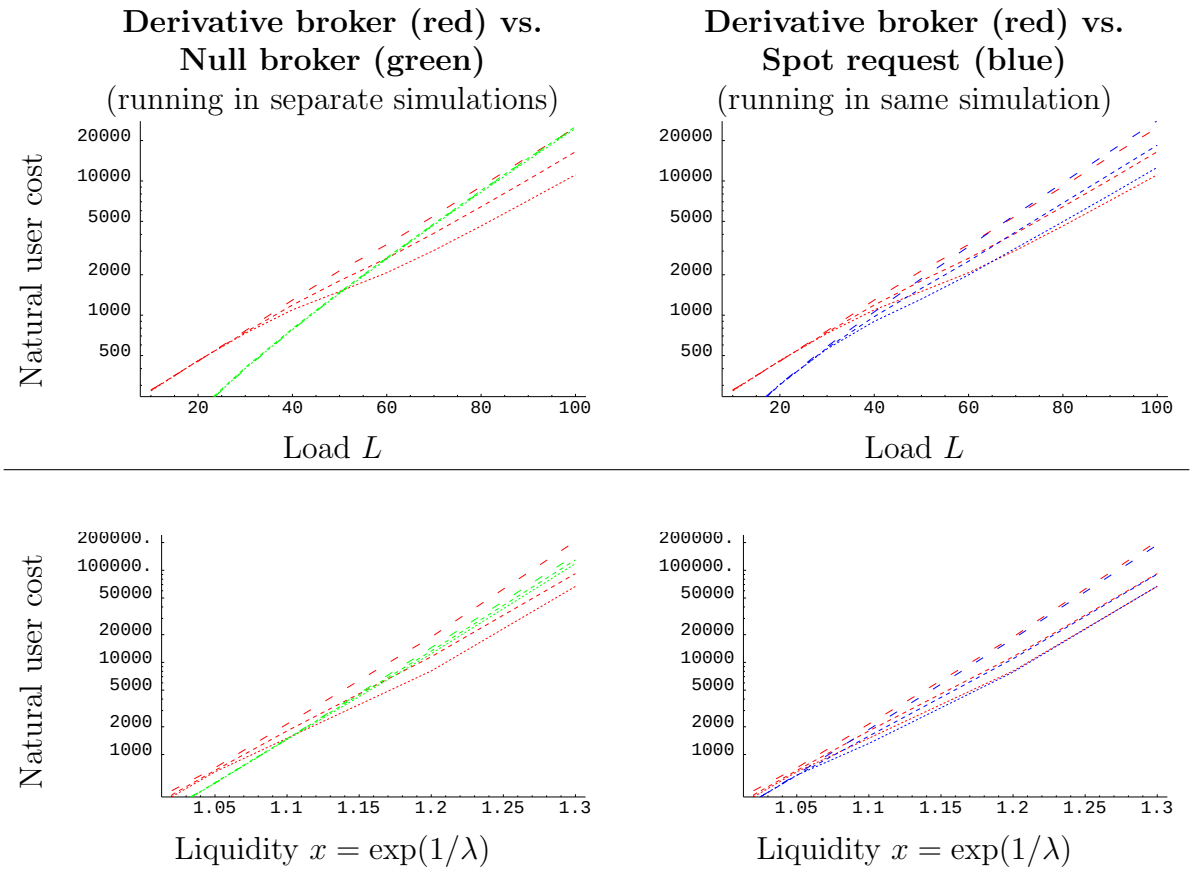

Figure 4: The two upper rows of fig. 3, the users' cost per request as a function of L (row one) and λ (row two), is plotted on a semi-log plot in natural currency units, i.e. the prices in fig. 3 are scaled by $\exp \frac{L}{\lambda}$. Color and dashing has same meaning as in fig. 2.

ratio of the latter schemes, as discussed in section 4.3. Since calls are rejected when the price is high, the average cost of an accepted call becomes low. This indicates that c_{spot} and c_{null} compute too high acceptance costs for low values of L, which causes the excessive rejection rate that is shown in row one of fig. 2. Since the relatively high end-user cost at low L results in higher admission rate, this does not contradict $H1$ and $H2$.

The end-users' spot request costs are even lower than the null broker request costs for small values of L. However, a user trading directly at the spot-market is exposed to the additional risk of having to pay more than c_{spot} for the service, due to fluctuations in the capacity prices. For small values of L, and other parameters at baseline values, about 7 percent of the spot-trading end-users exceed their budget, by on average about 30 percent.

As shown in fig. 4, for large values of L, the users' cost for derivative broker requests goes below the cost of both spot requests and null broker requests, while derivative broker requests maintain the lowest rejection rate. The performance of the derivative broker increases with q, the fraction of derivative broker requests.

In agreement with $H3$, increasing the alert time a decreases the users' cost for the derivative broker requests. It also decreases for the null broker requests, although with smaller slope. Similar to the spot requests loss behavior, changing a appears to change the users' cost for the spot requests, but again this is an effect of the spot requests free-riding on the derivative requests. In a simulation without derivative brokers, the spot requests do not benefit from increased alert time a.

Increasing the end-users' budget, by increasing ρ does, as expected, increase the average price paid per request, which is consistent with the decreased rejection rate observed on row four in fig. 2.

4.5 Broker costs

Fig. 5 shows the brokers' cost per request plotted against the data set parameters L, λ, a, and ρ, in the characteristic cost S^* scale. Fig. 6 shows the cost plotted in natural price units, i.e. rescaled by $\exp \frac{L}{\lambda}$. In these plots, a positive cost means that the broker is losing money on average, while a negative value means that money is earned on average.

The broker cost as a function of the load L shows that the null broker under-prices the point-to-point service, while the derivative broker over-prices it. Therefore, the null broker effectively "subsidizes" the requests, which

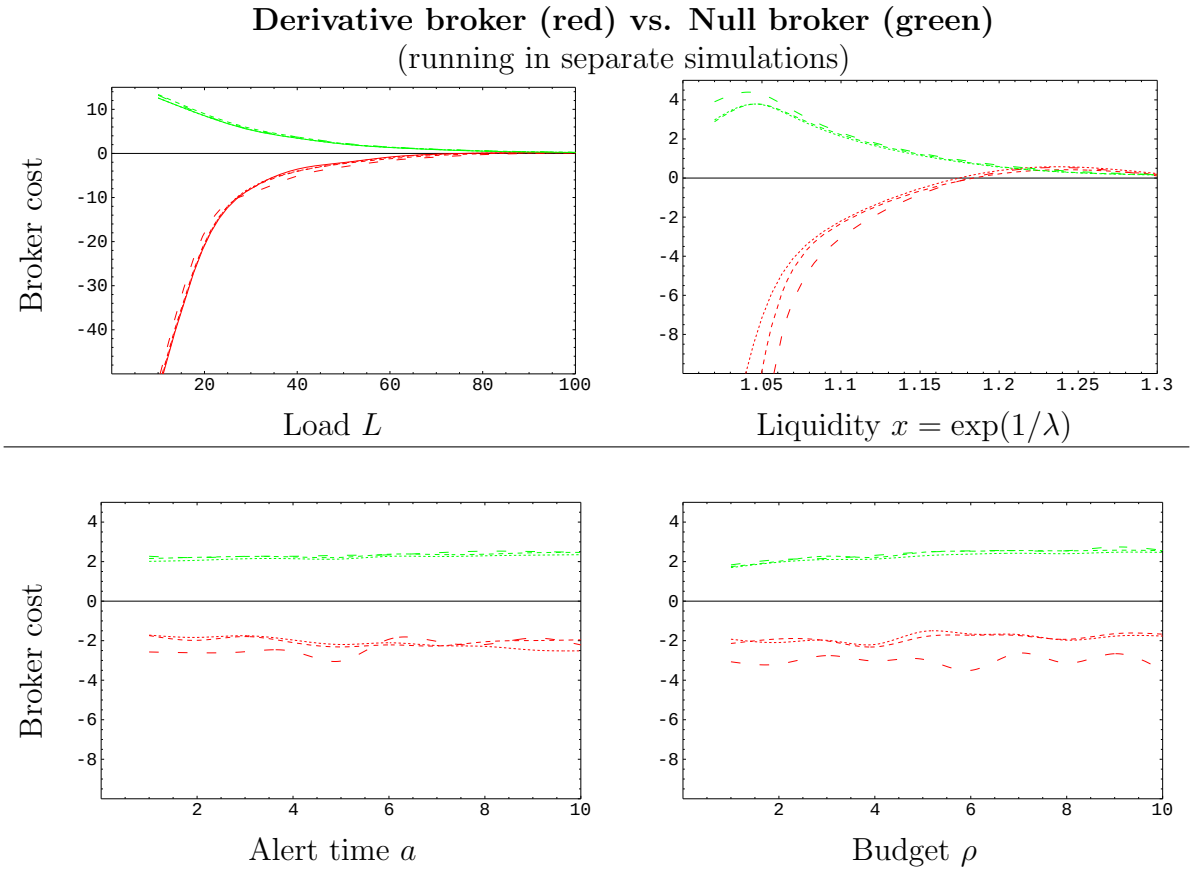

Figure 5: The brokers' cost per request is plotted on the characteristic cost S^* scale, for the four data sets. On the horizontal axis in row one are L and λ respectively, and and on row two, a and ρ. A positive broker cost means that the broker is losing money. Color and dashing has same meaning as in fig. 2. The wiggly lines are an artifact of fitting a cubic spline to noisy data .

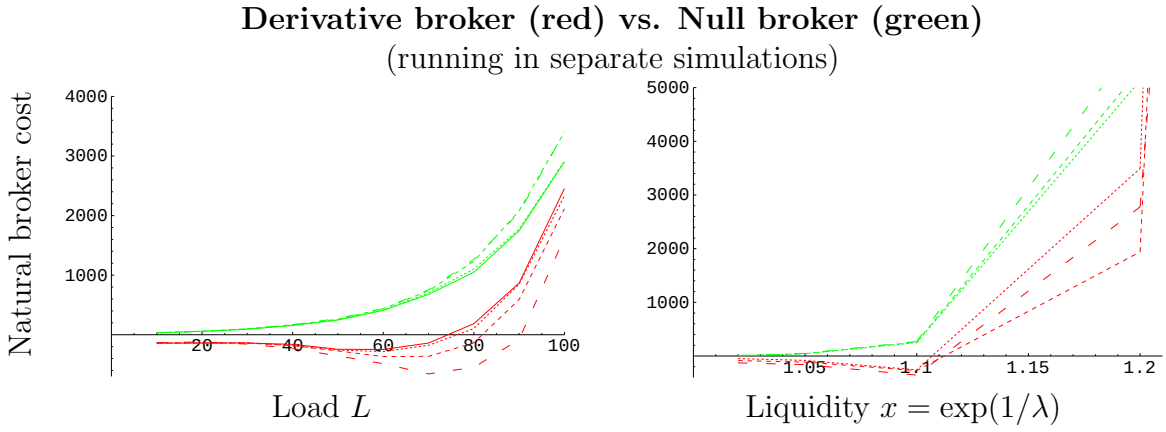

Figure 6: Here the upper row of fig. 5, the brokers' cost per request as a function of L (left) and λ (right), is plotted on a semi-log plot in natural currency units, i.e. the prices in fig. 5 are scaled by $\exp \frac{L}{\lambda}$. Color and dashing has the same meaning as in fig. 2.

results in a lower reject rate. The derivative broker overcharges, which to some extent explains the large increase in end-user cost for low values of L (see row one in fig. 3).

For large values of L and $\exp \frac{1}{\lambda}$, both types of broker requests under-price the point-to-point service. Extrapolating the curves outside the data set suggests that for very large values of the parameters, the derivative broker will surpass the null broker in underpricing the service. This is an artifact of the Monte Carlo method, which produces a larger relative errors when it samples wildly fluctuating prices.

The parameters a and ρ have in the investigated range no statistically significant impact on the brokers' cost per request.

4.6 Price variance

To study the effect on the price variance caused by different types of broker requests, the average price diffusion variance $\left\langle \sigma_{jj}^2 \right\rangle$ was computed, based at the estimate at $t = 999$, for distinct values of L, q, and *type*. Fig. 7 shows the ratio of the price volatility caused by derivative broker requests and null broker requests. For low values of L, the derivative broker request prices are more volatile than null broker prices, while for large L, the price volatility for derivative broker requests decreases below that of the null broker. The

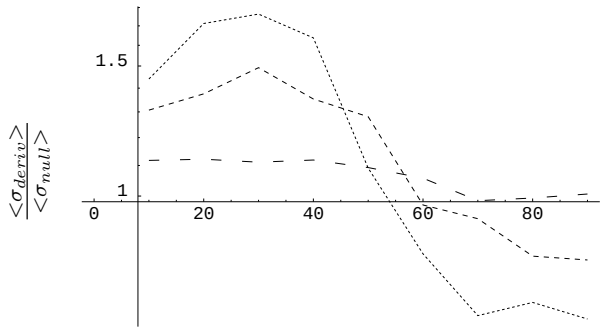

Figure 7: The average price variation for derivative broker requests, and for null broker requests is estimated at $t = 999$. The ratio is plotted against the load parameter L. Dashing has same meaning as in fig. 2.

volatility ratio decreases quite rapidly around $L = 50$, and thus appears to behave qualitatively similar to the user cost per request (fig. 4). This result is in agreement with $H4$ for $L > 50$ and in disagreement for $L < 50$.

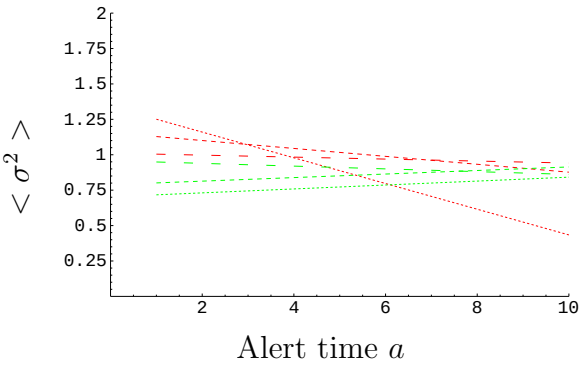

Figure 8: A linear fit of the average price variation is plotted against the alert time parameter a. Color denotes request type, and dashing denotes fraction of broker requests, as explained in sec. 4.2. The slope is significant only for derivative brokers when $q > 0.5$.

Fig. 8 shows a linear fit of the price diffusion variance $\left\langle \sigma_{jj}^2 \right\rangle$ plotted as a function of the alert time parameter a. The red densely dotted line shows that the derivative broker requests reduce the price variance, while the slope

is insignificant for the null broker. This is interpreted as evidence that the hedging strategy is able to use the information about future demand to reduce the price fluctuation, while the null broker trading strategy does not have this effect. It is consistent that the sparse dense lines have a smaller slope, since in those simulations, many of the requests were spot requests, which are not affected by the alert time. Thus, the graph in fig. 8 is consistent with $H4$.

This figure also provides an explanation to the spot requests' performance increase due to a, as shown in the right column on row three in fig. 2 and 3. The derivative brokers' damping effect on volatility is useful to the spot requests.

5 Conclusion

This paper reports on measurements of the effect of a broker trading mechanism for network capacity. The mechanism was tested in a computer network scenario where end-users face the task of composing virtual circuits, or paths, through a network, by buying capacity on several spot markets. The mechanism uses financial derivatives contracts and hedging techniques to allow a broker to trade and combine bundles of resources, when the single resources are traded on separate and independent markets. The derivatives based approach was compared to two other trading strategies that act as null hypotheses; a spot trading strategy that does not trade future demand, and a "null broker" strategy which trades future demand without performing any advance trading to reduce the risk. The performance was measured over a range of parameter values controlling the network's size in terms of router capacity, and user models. All simulation were run on one randomly generated network with 10 nodes.

The simulations showed that the derivative trading strategy admits more point-to-point request to the network than the null hypothesis trading strategies, even though the null broker "subsidized" admission by under-charging for the service it produced. This shows that it is the derivative broker's ability to trade future demand, and to balance the risk through hedging, that is responsible for its performance gain.

Brokers are useful in taking on the risk from the end-users. An end-user is more sensitive to over-spending than a broker, for whom profits and losses more quickly average out. The simulations showed that derivative brokers,

for an added cost, take on the end-users' trading risk, and reduce their risk of overspending from seven percent to zero. The results presented here show that the approach to trade network path components with the help of a derivative trading mechanism is feasible and has good effects on the network, compared to other simpler trading mechanisms. The main weakness of the derivatives based approach is the computational cost of Monte Carlo evaluation of the derivative price f and its partial derivatives. One potentially fruitful approach to improve on this is to approximate the derivative price function with a smooth interpolating function which can be evaluated rapidly. This is likely to be possible because the absolute values of partial derivatives of f are small, bounded by the number of nodes in the longest alternative path.

Acknowledgments

This work is part of the project Automatic Market Mechanisms for Resource Allocation in a Communication Network, nr 2001-4832 funded by Vinnova, the Swedish Agency for Innovation Systems. We want to thank, Carl-Gunnar Perntz at Ericsson Switchlab, and Anders Rockström at Skanova/Telia for comments and insightful discussions. Any errors and omissions are entirely ours.

References

[1] P. Ferguson and G. Huston. *Quality of Service: Delivering QoS on the Internet and in Corporate Networks*. John Wiley and Sons, January 1998.

[2] G. Ash. Traffic Engineering & QoS Methods for IP-, ATM-, & TDM-Based Multiservice Networks . IETF Internet Draft, October 2001.

[3] S. Chen and K. Nahrstedt. An overview of quality-of-service routing for the next generation high-speed networks: Problems and solutions, 1998.

[4] Errin W. Fulp, Maximilian Ott, Daniel Reininger, and Douglas S. Reeves. Paying for QoS: an optimal distributed algorithm for pricing network resources. In *Sixth International Workshop on Quality of Service (IWQoS'98)*, pages 75–84, 1998.

[5] Jakka Sairamesh, Donald F. Ferguson, and Yechiam Yemini. An Approach to Pricing, Optimal Allocation and Quality of Service Provisioning In High-Speed Packet Networks. In *Proc. of IEEE INFOCOM*, pages 1111–1119, Boston, MA, April 1995. IEEE.

[6] Alok Gupta, Dale O. Stahl, and Andrew B. Whinston. An Economic Approach to Network Computing with Priority Classes. *J. of Organizational Computing and Electronic Commerce*, 6:71–95, 1996.

[7] Panagiotis Thomas, Demosthenis Teneketzis, and Jeffrey K. MacKie-Mason. A Market-Based Approach to Optimal Resource Allocation in Integrated-Services Connection-Oriented Networks. *Operations Research*, 50(4), July-August 2001.

[8] S. Chen and K. Nahrstedt. On Finding Multi-constrained Paths. In *ICC'98*, pages 874 –879. IEEE, 1998.

[9] G. Cheliotis. A Market-Based Model of Bandwidth and a New Approach to End-to-End Path Computation with Quality Guarantees. Technical report, IBM Research, Zurich, November 2000.

[10] M. S. Miller, D. Krieger, N. Hardy, C. Hibbert, and E. D. Tribble. An Automated Auction in ATM Network Bandwidth. In Scott H. Clearwater, editor, *Market-based Control, A Paradigm for Distributed Resource Allocation*. World Scientific, Palo Alto, CA, 1996.

[11] Alok Gupta, Dale O. Stahl, and Andrew B. Whinston. A Stochastic Equilibrium Model of Internet Pricing. *J. of Economic Dynamics and Control*, 21(4–5):697–722, 1997.

[12] Alok Gupta, Dale O. Stahl, and Andrew B. Whinston. The economics of network management. *Communications of the ACM*, 42(9):57–63, 1999.

[13] Manoj Parameswaran, Jan Stallaert, and Andrew B. Whinston. A market-based allocation mechanism for the DiffServ framework. *Decision Support Systems*, 31(3):351–361, 2001.

[14] Jeffrey K. MacKie-Mason and Hal R. Varian. Pricing the Internet. In Brian Kahin and James Keller, editors, *Public Access to the Internet*, pages 269–314. MIT Press, Cambridge, MA, 1995.

[15] Richard J. Gibbens and Frank P. Kelly. Resource pricing and the evolution of congestion control. *Automatica*, 35(12):1969–1985, December 1999.

[16] Richard J. Gibbens and Frank P. Kelly. Distributed connection acceptance control for a connectionless network. In P. B. Key and D. G. Smith, editors, *Proceedings of 16th International Teletraffic Congress*, pages 941–952. Elsevier, June 1999.

[17] Frank Kelly. Mathematical modelling of the Internet. In B. Engquist and W. Schmid, editors, *Mathematics Unlimited - 2001 and Beyond*, pages 685 – 702. Springer-Verlag, Berlin, 2001.

[18] Peter Marbach. Pricing Differentiated Services Networks: Bursty Traffic. In *Proc. of INFOCOM*. IEEE, 2001.

[19] Steven Blake, David Black, Mark Carlson, Elwyn Davies, Zheng Wang, and Walter Weiss. RFC 2475: An Architecture for Differentiated Services. IETF, October 1998.

[20] Nemo Semret and Aurel A. Lazar. Spot and Derivative Markets in Admission Control. In P. B. Key and D. G. Smith, editors, *Proceedings of 16th International Teletraffic Congress*, pages 925–941. Elsevier, June 1999.

[21] Costas Courcoubetis, Antonis Dimakis, and Martin I. Reiman. Providing Bandwidth Guarantees over a Best-effort Network: Call-admission and Pricing. In *INFOCOM*, pages 459–467, 2001.

[22] Chris Kenyon and Giorgos Cheliotis. Stochastic models for telecom commodity prices. *Computer Networks*, 36(5–6):533–555, August 2001.

[23] Girogos Cheliotis. *Structure and Dynamics of Bandwidth Markets*. PhD thesis, National Technical University of Athens, November 2001.

[24] Jussi Keppo. Pricing of bandwidth derivatives under network arbitrage condition. Working paper, University of Michigan, August 2001.

[25] Lars Rasmusson and Gabriel Paues. Network components for market-based network admission and routing . Technical Report SICS--T2002-22--SE, SICS, Stockholm, Sweden, 2002.

[26] Lars Rasmusson and Erik Aurell. A Price Dynamics in Bandwidth Markets for Point-to-point Connections. Technical Report SICS--T2001-21--SE, SICS, Stockholm, Sweden, 2001.

[27] Lars Rasmusson. Evaluating Resource Bundle Derivatives for Multi-Agent Negotiation of Resource Allocation. In *E-Commerce Agents: Marketplace Solutions, Security Issues, and Supply and Demand*, volume 2033 of *Lecture Notes in Artificial Intelligence (LNAI)*. Springer Verlag, Berlin, 1999.

[28] Lars Rasmusson. Evaluating the CDF for m weighted sums of n correlated lognormal random variables. In *Proc. of the 8th Int. Conf. on Computing in Economics and Finance*, Aix-en-Provence, France, June 2002. Soc. for Computational Economics.

[29] Stephen Wolfram. *The Mathematica Book*. Cambridge University Press, 1996.

[30] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C, The Art of Scientific Computing*. Cambridge University Press, 2nd edition, 1992.

[31] E. L. Lehmann. *Nonparametric Statistical Methods Based on Ranks*. McGraw-Hill, New York, 1975.